Patent
110630-016

UNITED STATES PATENT APPLICATION
FOR

IMAGE PROCESSING AND ANALYSIS FRAMEWORK

INVENTORS:

OLE EICHHORN
ALLEN OLSON

PREPARED BY:
PROCOPIO, CORY, HARGREAVES & SAVITCH LLP
530 B STREET, SUITE 2100
SAN DIEGO, CALIFORNIA 92101-4469

IMAGE PROCESSING AND ANALYSIS FRAMEWORK

Related Application

[01] The present application claims priority to United States provisional patent application serial number 60/451,081 entitled DEVELOPMENT AND TESTING PLATFORM AND RUNTIME ENVIRONMENT FOR IMAGE PROCESSING filed on February 28, 2003, which is incorporated herein by reference in its entirety.

Background

1. Field of the Invention

[02] The present invention generally relates to virtual microscopy and more specifically relates to the processing and analysis of virtual slides.

2. Related Art

[03] In the growing field virtual microscopy, the first challenges to be overcome were related to the digital imaging of microscope slides ("scanning"). Conventional image tiling is one approach that is widely prevalent in the virtual microscopy industry. The image tiling approach to scanning microscope slides employs a square or rectangular camera called a fixed area charge coupled device ("CCD"). The CCD camera takes hundreds or thousands of individual pictures ("image tiles") of adjacent areas on the microscope slide. Then the thousands of image tiles are each separately stored as a bitmap ("bmp") or a JPEG ("jpg") file on a computer. An index file is also required in order to identify the name of each image tile and its relative location in the overall image. As would be expected, the taking of thousands of individual pictures and storing each picture as an image tile along with creation of the index files takes a significantly long time. A conventional image tiling approach is described in U.S. Patent 6,101,265. Although slow and cumbersome, conventional image tiling solutions did succeed in scanning microscope slides to create a virtual slide.

[04] Once the virtual slide was present in a computer system, computer assisted image analysis became possible. Two significant drawbacks of processing image tiles are the computational expense of aligning tiles and correlating overlaps, and the presence of image artifacts along the seams between tiles. These problems each prevented practical application of automated image analysis to virtual slide images. It has also proved difficult to maintain accurate focus for each of thousands of tiles in a virtual slide produced in this way, reducing image quality.

[05] A radical change in the virtual microscopy field has recently been developed by Aperio Technologies, Inc. that uses a new line scanning system to create a virtual slide in minutes. It also creates the virtual slide as a single TIFF file. This revolutionary line scanning system employs a line scan camera (i.e., called a linear-array detector) in conjunction with specialized optics, as described in U.S. Patent Number _____ entitled "Fully Automatic Rapid Microscope Slide Scanner," which is currently being marketed under the name ScanScope®.

[06] In addition to rapid data capture and creating a single file virtual slide, the line scanning system also benefits from several advantages that ensure consistently superior imagery data. First, focus of the linear array can be adjusted from one scan line to the next, while image tiling systems are limited to a single focal plane for an entire image tile. Second, because the linear array sensor in a line scanning system is one-dimensional (i.e., a line), there are no optical aberrations along the scanning axis. In an image tiling system, the optical aberrations are circularly symmetric about the center of the image tile. Third, the linear array sensor has a complete (100%) fill factor, providing full pixel resolution (8 bits per color channel), unlike color CCD cameras that lose spatial resolution because color values from non-adjacent pixels must be interpolated (e.g., using a Bayer Mask).

[07] The creation of a single file virtual slide is an enormously significant improvement. Managing a single image file for a virtual slide requires significantly less operating system overhead than the management of thousands of individual image tiles

and the corresponding index file. Additionally, alignment of component images may be computed once, then re-used many times for automated processing.

[08] Therefore, introduction of the superior line scanning system for creating single file virtual slides has created a need in the industry for efficient virtual slide image analysis systems and methods that meet the unique needs imposed by the new technology.

Summary

[09] A system and method for processing and analyzing virtual microscopy digital images ("virtual slides") is provided. The system comprises an algorithm server that maintains or has access to a plurality of image processing and analysis routines. The algorithm server additionally has access to a plurality of virtual slides. The algorithm server executes a selected routine on an identified virtual slide and provides the resulting data. The virtual slide can be accessed locally or remotely across a network. Similarly, the image processing routines can be obtained from local storage or across a network, or both. Advantageously, certain common sub-routines may be stored locally for inclusion in other local or remotely obtained routines.

[10] Support for multiple users to access the virtual slide sever and execute image analysis routines is provided through a monitor process that allows multiple inbound connections and routes those connections to the appropriate module within the server for processing. Additionally, the monitor process may also restrict access to viewing or processing of virtual slides by enforcing various security policies such as limitations on source network addresses, username and password authentication, and session timeouts. These and other variations in access to viewing and analyzing images provide a rich diversity in access levels that allow sharing of virtual slides and demonstrations of image processing algorithms.

Brief Description of the Drawings

[11] The details of the present invention, both as to its structure and operation, may be gleaned in part by study of the accompanying drawings, in which like reference numerals refer to like parts, and in which:

[12] **Figure 1** is a network diagram illustrating an example system for image processing and analysis according to an embodiment of the present invention;

[13] **Figure 2** is a block diagram illustrating an example algorithm server according to an embodiment of the present invention;

[14] **Figure 3** is a flow diagram illustrating an example process for executing an image processing algorithm according to an embodiment of the present invention;

[15] **Figure 4** is a flow diagram illustrating an example process for creating an image processing macro according to an embodiment of the present invention;

[16] **Figure 5** is a flow diagram illustrating an example process for importing a remote image processing algorithm according to an embodiment of the present invention;

[17] **Figure 6** is a flow diagram illustrating an example process for remotely executing an image processing algorithm according to an embodiment of the present invention; and

[18] **Figure 7** is a block diagram illustrating an exemplary computer system as may be used in connection with various embodiments described herein.

Detailed Description

[19] Certain embodiments as disclosed herein provide a framework for processing and analysis of virtual slide images. The system comprises an algorithm server that executes image processing instructions (referred to herein as "algorithms," "routines," and "sub-routines") on virtual slide images or sub-regions of a virtual slide image. For example, one method disclosed herein allows a user to identify a virtual slide image (or a sub-region thereof) and an algorithm to be used in the processing and analysis of the image. The server then executes the algorithm to process and analyze the image. The results may be provide to the screen, to a file, to a database, or otherwise presented, captured and/or recorded. Certain parameters may also be provided by the user or obtained from a

data file corresponding to the particular algorithm to constrain the processing and analysis called for in the algorithm.

[20] After reading this description it will become apparent to one skilled in the art how to implement the invention in various alternative embodiments and alternative applications. However, although various embodiments of the present invention will be described herein, it is understood that these embodiments are presented by way of example only, and not limitation. As such, this detailed description of various alternative embodiments should not be construed to limit the scope or breadth of the present invention as set forth in the appended claims.

[21] Fig. 1 is a network diagram illustrating an example system 10 for image processing and analysis according to an embodiment of the present invention. In the illustrated embodiment, the system 10 comprises an algorithm server 20 that is communicatively linked with one or more remote users 50 and one or more remote image servers 60 via a network 80. The algorithm server is configured with a data storage area 40 and a plurality of local image files 30. The data storage area preferably comprises information related to the processing of digital image files, for example it may store certain analysis routines, parameters, and procedural lists of analysis routines and associated parameters ("macros"), among other types of data. The local image files 30 and remote image files 70 are preferably virtual slides of the type created by the ScanScope® Microscope Slide Scanner developed by Aperio Technologies, Inc.

[22] The remote image server 60 is preferably configured with a data storage area having a plurality of image files 70 that are considered remote from the algorithm server. In one embodiment, the algorithm server 20 may access the remote virtual slide images 70 via the network 80. The remote user 50 may comprise a web browser, an imagescope viewer, a scanscope console, an algorithm framework, or some other client application or front end that facilitates a user's interaction with the algorithm server 20. The network 80 can be a local area network ("LAN"), a wide area network ("WAN"), a private network, public network, or a combination of networks such as the Internet.

[23] Fig. 2 is a block diagram illustrating an example algorithm server 20 according to an embodiment of the present invention. In the illustrated embodiment, the algorithm server 20 comprises an execution manager 100, an image handler 110, a user interface 120, a remote user manager 130, a reporting manager 140, and a security daemon 150. The algorithm server is also configured with a data storage area 40 and a plurality of local image files 30 and a plurality of remote image files 70. Preferably, the local and remote image files are virtual slides.

[24] The execution manager 100 handles the process of executing an algorithm to conduct image analysis or other types of analysis on a virtual slide. The execution manager 100 can be in communication with the other components of the algorithm server in order to process the image analysis requests from one or more local or remote users. For example, the execution manager 100 is configured to receive an instruction from a user to run a particular algorithm on a certain virtual slide. The execution manager 100 is also configured to collect parameter data from the user that will be used by the algorithm during execution. The parameters may define a sub-region of the virtual slide to be processed by the algorithm and/or may define certain threshold values or provide other data elements to constrain the algorithm during execution.

[25] The execution manager 100 is communicatively coupled with the image handler 110 in order to obtain the virtual slide (or portion thereof) for analysis and processing pursuant to the particular algorithm. Because virtual slides are such large files (10-15 GB uncompressed), a specialized image handler 110 is employed to efficiently and quickly obtain image data for processing. Advantageously, the image handler 110 can obtain image data from virtual slides that are stored either locally (image files 30) or remotely (image files 70). Additionally, the image handler 110 provides the virtual slide image in a standard format by decompressing the stored virtual slide image from various compression formats including JPEG, JPEG2000, and LZW formats.

[26] Another function of the image handler 110 is to provide the image data from a virtual slide at the proper level of magnification. For example, an image may be stored in at a native resolution of 40X but a resolution of 20X is called for by the algorithm. The

image handler 110 can downsample the native resolution and deliver the image data at a resolution of 20X to the execution manager 100. Such ability provides a significant advantage in speed for algorithms that initially process an image at a low resolution where objects of interest are detected and subsequently process the image at a high resolution where the analysis of those sub-regions containing the identified objects of interest is carried out. For example, the amount of image data to be processed in a 40X image is four times the amount of image data to be processed in a 20X image, so an algorithm which processes a 40X image at a resolution of 20X can run four times faster.

[27] The user interface 120 preferably provides the user with a simple and easy to use format for interacting with the execution manager 100 in order to identify the algorithm to execute and the virtual slide to be analyzed. Additionally the user interface 120 can efficiently collect parameter data from a user prior to execution of an algorithm. The user interface 120 also allows a user to create a macro comprising a plurality of algorithms and associated parameters.

[28] The remote user manager 130 preferably manages the connection and communication with a user that is accessing the algorithm server 20 through a network connection. The remote user manager 130 is also configured to receive requests from network based users and programs, and to process those requests in real time or as a scheduled batch.

[29] The reporting manager 140 is preferably configured to receive output and processing results from the execution manager 100 as an algorithm executes and generates data or other output. The reporting manager may also access output files after an algorithm is finished processing and then restructure the data in the output file into a standard report format. Additional reporting capabilities may also be provided by the reporting manager 140, as will be understood by those having skill in the art.

[30] The security daemon 150 advantageously handles image processing requests that originate from a network based user or program. The security daemon 150 receives all incoming network requests and examines those requests to determine if they are seeking to process an image with an algorithm. If so, the security daemon 150 is configured to

validate the request and if the request is valid, then the security daemon 150 passes the request off to the execution manager 150 for processing.

[31] In one embodiment, the data storage area 40 may contain a plurality of algorithms that can be executed in order to analyze virtual slide images. Additionally, the data storage area 40 may also contain a plurality of sub-routines that are commonly performed or often included in image processing algorithms. Advantageously, these common sub-routines can be dynamically linked into an algorithm at runtime so that the algorithm development effort is simplified. Additionally, the data storage area 40 may also comprise a plurality of macros, where a macro comprises a linear or parallel sequence of processing images with algorithms to achieve a desired analysis. A macro may also comprise parameter data to define subregions of images where appropriate and provide variables to constrain the image processing.

[32] Fig. 3 is a flow diagram illustrating an example process for executing an image processing algorithm according to an embodiment of the present invention. Initially, in step 200, the execution manager receives an image selection. The image selection can be for a particular virtual slide, or an identified sub-region thereof. Next, in step 210, the execution manager receives a selection for the algorithm to be run. There may in fact be more than one algorithm, or the execution manager may receive the selection of a macro that comprises several algorithms. In step 220, the execution manager receives parameter data necessary to run the algorithm(s). Advantageously, the execution manager may query the algorithm or check a corresponding data file to determine what parameter data will be required to run the algorithm(s). Finally, in step 230, after the image has been selected and the algorithm selected and the parameter data provided, the execution manager runs the algorithm and preferably provides any output to an output file, the screen, a database, or other display or storage facility.

[33] Fig. 4 is a flow diagram illustrating an example process for creating an image processing macro according to an embodiment of the present invention. Initially, in step 300, the execution manager receives a selection for the first algorithm to be run as part of the macro. Next, in step 310, the execution manager receives the parameter data that

corresponds to the selected algorithm. In some cases, all of the parameter may not be provided when the macro is created. Advantageously, the execution manager can collect a partial set of parameter data and then when the macro is run, the execution manager can prompt the user for the needed additional parameter data. For example, the user or program requesting the macro be run will also have to identify the virtual slide to be processed, and optionally the subregion of the image on which the algorithm will operate.

[34] Once the algorithm selection and the parameter data have been collected, the user is prompted to determine in step 320 if there are more algorithms to be included in the macro. If so, then the process loops back to collect the algorithm selection and parameter data as described immediately above. If the macro definition is complete, then in step 330 the macro is stored in persistent data storage for later retrieval and execution.

[35] Fig. 5 is a flow diagram illustrating an example process for importing a remote image processing algorithm according to an embodiment of the present invention. In the first step 400, the execution manager receives the identification of an algorithm that is located on another server or computer that is remotely located and accessible via the network. Once the remote algorithm has been identified, in step 410 the execution manager receives any algorithm attributes and an identification of the parameters required by the algorithm during execution. These attributes and parameter requirements are preferably stored in a fashion that they correspond to the algorithm so that at a later time the execution manager may retrieve this information prior to execution of the algorithm, which will facilitate the collection of the parameter data prior to execution. Finally, in step 420, the new algorithm is stored along with its associated parameters and attributes.

[36] Fig. 6 is a flow diagram illustrating an example process for remotely executing an image processing algorithm according to an embodiment of the present invention. Initially, in step 450, the execution manager receives a request from a remote user or program. Advantageously, the remote user may be an automated program that is developed to automate certain types of image processing on virtual slides. Upon receipt of the request, in step 460, the execution manager parses the request to obtain information

such as the particular algorithm to run, the identification of the virtual slide (and optionally the sub-region thereof), as well as the parameter data.

[37] Next, in step 470, the execution manager loads the algorithm and then also loads the image in step 480. At this point, the execution manager runs the algorithm to process the identified virtual slide image, using the parameter data received in the request as necessary and in accordance with the particular algorithm. Any output generated by the running of the algorithm can advantageously be sent to a local output file or database, or it can be collected and the distributed remotely or locally or stored, as shown in step 492.

[38] Virtual slide images are very large, and it may be impractical to load an entire image in step 480. In such cases steps 480 and 490 can be repeated iteratively for multiple consecutive sub-regions of the virtual slide image. Depending on the nature of the algorithm, it may be necessary or desirable to overlap the processed sub-regions, and then adjust the algorithm results. Accordingly, in step 494, the execution manager determines if there is more image data or additional image sub-regions to process. If so, the process loops back to step 480 where the next sub-region of the image is loaded. If there are no more sub-regions to process, then the execution manager can end the image processing, as illustrated in step 496.

[39] Furthermore, some algorithms may benefit from multiple "passes" or recursive analysis/processing on an image. For example, a first pass (execution of the image processing instructions) can be made at low resolution (20X) to identify sub-regions of the image which require further analysis. Then a second pass (execution of the image processing instructions) can be made at high resolution (40X) to process and analyze just those sub-regions identified in the first pass. Advantageously, the algorithm results would reflect both passes (the sub-region identification and the output of the processing of the sub-regions).

[40] Fig. 7 is a block diagram illustrating an exemplary computer system 550 that may be used in connection with the various embodiments described herein. For example, the computer system 550 may be used in conjunction with an algorithm server, a remote

image server or a remote user station. However, other computer systems and/or architectures may be used, as will be clear to those skilled in the art.

[41] The computer system 550 preferably includes one or more processors, such as processor 552. Additional processors may be provided, such as an auxiliary processor to manage input/output, an auxiliary processor to perform floating point mathematical operations, a special-purpose microprocessor having an architecture suitable for fast execution of signal processing algorithms (e.g., digital signal processor), a slave processor subordinate to the main processing system (e.g., back-end processor), an additional microprocessor or controller for dual or multiple processor systems, or a coprocessor. Such auxiliary processors may be discrete processors or may be integrated with the processor 552.

[42] The processor 552 is preferably connected to a communication bus 554. The communication bus 554 may include a data channel for facilitating information transfer between storage and other peripheral components of the computer system 550. The communication bus 554 further may provide a set of signals used for communication with the processor 552, including a data bus, address bus, and control bus (not shown). The communication bus 554 may comprise any standard or non-standard bus architecture such as, for example, bus architectures compliant with industry standard architecture ("ISA"), extended industry standard architecture ("EISA"), Micro Channel Architecture ("MCA"), peripheral component interconnect ("PCI") local bus, or standards promulgated by the Institute of Electrical and Electronics Engineers ("IEEE") including IEEE 488 general-purpose interface bus ("GPIB"), IEEE 696/S-100, and the like.

[43] Computer system 550 preferably includes a main memory 556 and may also include a secondary memory 558. The main memory 556 provides storage of instructions and data for programs executing on the processor 552. The main memory 556 is typically semiconductor-based memory such as dynamic random access memory ("DRAM") and/or static random access memory ("SRAM"). Other semiconductor-based memory types include, for example, synchronous dynamic random access memory

("SDRAM"), Rambus dynamic random access memory ("RDRAM"), ferroelectric random access memory ("FRAM"), and the like, including read only memory ("ROM").

[44] The secondary memory 558 may optionally include a hard disk drive 560 and/or a removable storage drive 562, for example a floppy disk drive, a magnetic tape drive, a compact disc ("CD") drive, a digital versatile disc ("DVD") drive, etc. The removable storage drive 562 reads from and/or writes to a removable storage medium 564 in a well-known manner. Removable storage medium 564 may be, for example, a floppy disk, magnetic tape, CD, DVD, etc.

[45] The removable storage medium 564 is preferably a computer readable medium having stored thereon computer executable code (i.e., software) and/or data. The computer software or data stored on the removable storage medium 564 is read into the computer system 550 as electrical communication signals 578.

[46] In alternative embodiments, secondary memory 558 may include other similar means for allowing computer programs or other data or instructions to be loaded into the computer system 550. Such means may include, for example, an external storage medium 572 and an interface 570. Examples of external storage medium 572 may include an external hard disk drive or an external optical drive, or and external magneto-optical drive.

[47] Other examples of secondary memory 558 may include semiconductor-based memory such as programmable read-only memory ("PROM"), erasable programmable read-only memory ("EPROM"), electrically erasable read-only memory ("EEPROM"), or flash memory (block oriented memory similar to EEPROM). Also included are any other removable storage units 572 and interfaces 570, which allow software and data to be transferred from the removable storage unit 572 to the computer system 550.

[48] Computer system 550 may also include a communication interface 574. The communication interface 574 allows software and data to be transferred between computer system 550 and external devices (e.g. printers), networks, or information sources. For example, computer software or executable code may be transferred to computer system 550 from a network server via communication interface 574. Examples

of communication interface 574 include a modem, a network interface card ("NIC"), a communications port, a PCMCIA slot and card, an infrared interface, and an IEEE 1394 fire-wire, just to name a few.

[49] Communication interface 574 preferably implements industry promulgated protocol standards, such as Ethernet IEEE 802 standards, Fiber Channel, digital subscriber line ("DSL"), asynchronous digital subscriber line ("ADSL"), frame relay, asynchronous transfer mode ("ATM"), integrated digital services network ("ISDN"), personal communications services ("PCS"), transmission control protocol/Internet protocol ("TCP/IP"), serial line Internet protocol/point to point protocol ("SLIP/PPP"), and so on, but may also implement customized or non-standard interface protocols as well.

[50] Software and data transferred via communication interface 574 are generally in the form of electrical communication signals 578. These signals 578 are preferably provided to communication interface 574 via a communication channel 576. Communication channel 576 carries signals 578 and can be implemented using a variety of communication means including wire or cable, fiber optics, conventional phone line, cellular phone link, radio frequency (RF) link, or infrared link, just to name a few.

[51] Computer executable code (i.e., computer programs or software) is stored in the main memory 556 and/or the secondary memory 558. Computer programs can also be received via communication interface 574 and stored in the main memory 556 and/or the secondary memory 558. Such computer programs, when executed, enable the computer system 550 to perform the various functions of the present invention as previously described.

[52] In this description, the term "computer readable medium" is used to refer to any media used to provide computer executable code (e.g., software and computer programs) to the computer system 550. Examples of these media include main memory 556, secondary memory 558 (including hard disk drive 560, removable storage medium 564, and external storage medium 572), and any peripheral device communicatively coupled with communication interface 574 (including a network information server or other

network device). These computer readable mediums are means for providing executable code, programming instructions, and software to the computer system 550.

[53] In an embodiment that is implemented using software, the software may be stored on a computer readable medium and loaded into computer system 550 by way of removable storage drive 562, interface 570, or communication interface 574. In such an embodiment, the software is loaded into the computer system 550 in the form of electrical communication signals 578. The software, when executed by the processor 552, preferably causes the processor 552 to perform the inventive features and functions previously described herein.

[54] Various embodiments may also be implemented primarily in hardware using, for example, components such as application specific integrated circuits ("ASICs"), or field programmable gate arrays ("FPGAs"). Implementation of a hardware state machine capable of performing the functions described herein will also be apparent to those skilled in the relevant art. Various embodiments may also be implemented using a combination of both hardware and software.

[55] While the particular systems and methods herein shown and described in detail are fully capable of attaining the above described objects of this invention, it is to be understood that the description and drawings presented herein represent a presently preferred embodiment of the invention and are therefore representative of the subject matter which is broadly contemplated by the present invention. It is further understood that the scope of the present invention fully encompasses other embodiments that may become obvious to those skilled in the art and that the scope of the present invention is accordingly limited by nothing other than the appended claims.